Robustness Envelopes for Temporal Plans

Michael Cashmore[1] Alessandro Cimatti[2] Daniele Magazzeni[1]
Andrea Micheli[2] Parisa Zehtabi[1]

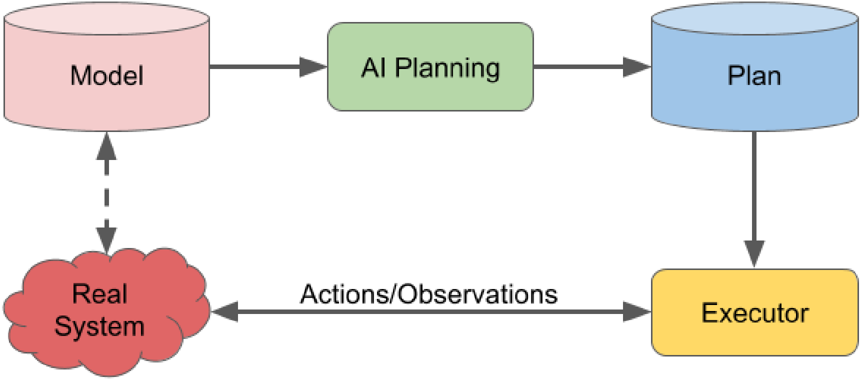
[1] Department of Informatics, King's College London, UK
[2] Embedded Systems Unit, Fondazione Bruno Kessler, Italy

10th November 2018

AAAI 2019, Honolulu, HA, USA

Temporal Planning and Execution

Plans generated from an automated planner need to be executed in the real world, that might be not aligned with the model used for planning

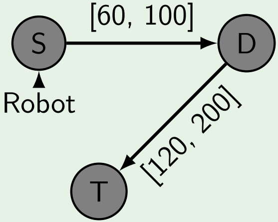


Classic Solution: STN Plans and Flexibility

Leave some freedom to the executor to reschedule actions by constraining relevant time-points instead of fixing them

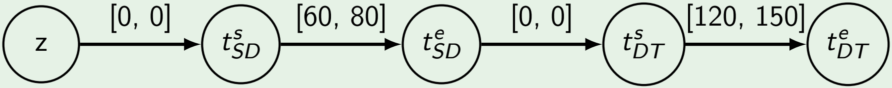
Example

Simple navigation planning problem:



- Robot must collect some data in D and transmit it T
- battery is drained at a constant rate of 0.4% per time unit

STN plan:



Outline

A First Problem: Validation

An STN plan allows several (often infinite) executions. We need to ensure that each of these is:

1. executable (action conditions are satisfied)
2. resource-valid (resource constraints are always satisfied)
3. goal-reaching

Contribution #1

Technique to automatically validate STN plan for action-based planning languages

Robustness Envelopes

Problem: understand and generalize plan applicability when some quantities (e.g. durations, consumption rates, ...) differ from the model

Input

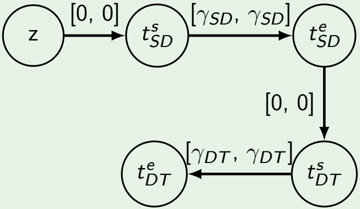
1. a set of numeric parameters
2. a planning problem that may use some parameters
3. an STN plan that may use some parameters

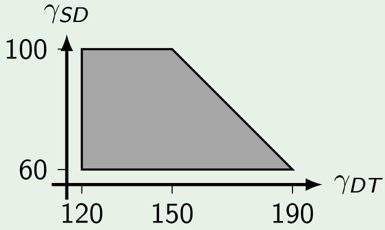
Output

The region of all possible parameter evaluation that keeps the STN plan valid for the planning problem

Robustness Envelopes

Example



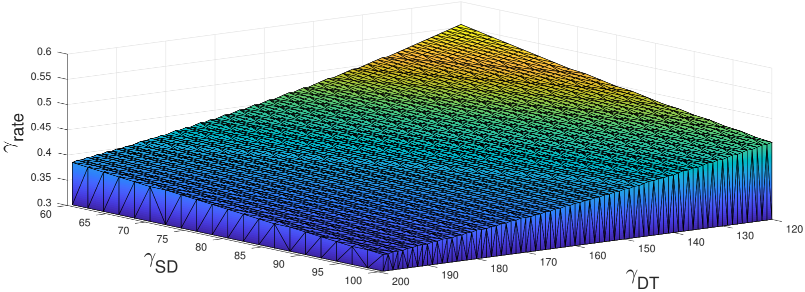


Contribution #2

Technique to automatically synthesize Robustness Envelopes given a parametric planning problem (in PDDL 2.1 with continuous resources) and an STN plan

More Complex Envelopes

In the previous example, assume that action uniformly consume battery at a rate γ_{rate}



Studying the envelopes allows understanding of parameter inter-dependencies

Outline

1. Problem Statements
2. SMT-based techniques
3. Experiments
4. Conclusion

Satisfiability Modulo Theory (SMT)

Overall Idea

Leverage SMT framework to uniformly, logically encode and solve the validation and synthesis problems

SMT is the problem of deciding the satisfiability of a first-order formula expressed in a given (decidable) theory T.

A formula ϕ is satisfiable if there exists a first-order interpretation μ such that $\mu \models \phi$.

Example

$\phi \doteq (x > 2) \wedge (x < 8) \wedge ((x < 1) \vee (x > 7))$

- Is satisfiable in the Theory of Real Arithmetic because $\{x \doteq 7.5\} \models \phi$
- Is unsatisfiable in the Theory of Integer Arithmetic

The SMT Encoding: Validity

Components

1. enc^{π}_{tn}: encodes the temporal constraints imposed by π limiting the possible orderings of time points.
2. enc^{π}_{eff} encodes the effects of each time point on the fluents and predicates
3. enc^{π}_{proofs} encodes the validity properties of the plan, namely:
   - conditions of each executed action are satisfied
   - the goal is reached
   - ϵ-separation constraint imposed by PDDL 2.1 is respected.

Theorem (STN Plan Validity)

π is a valid plan for $\mathcal{P}$ if:

1. $enc^{\pi}_{tn} \wedge enc^{\pi}_{eff}$ *is satisfiable*
2. $enc^{\pi}_{tn} \wedge enc^{\pi}_{eff} \rightarrow enc^{\pi}_{proofs}$ *is valid*

The SMT Encoding: Synthesis

Add parameters variables ($\bar{\Gamma}$) to the formulae: $enc_{tn}^{\pi_\Gamma}$, $enc_{eff}^{\pi_\Gamma}$ and $enc_{proofs}^{\pi_\Gamma}$

Robustness Envelope Synthesis

$$\rho(\bar{\Gamma}) \doteq \exists \bar{X}.(enc_{tn}^{\pi_\Gamma} \wedge enc_{eff}^{\pi_\Gamma}) \wedge \forall \bar{X}.((enc_{tn}^{\pi_\Gamma} \wedge enc_{eff}^{\pi_\Gamma}) \rightarrow enc_{proofs}^{\pi_\Gamma})$$

The models of $\rho(\bar{\Gamma})$ are all and only the parameter values that make the plan valid for the problem.

$\rho(\bar{\Gamma})$ encodes the Robustness Envelope!

Dealing with Quantifiers

The formula $\rho(\bar{\Gamma})$ contains quantifiers, so it is hard to exploit for plan generalization and analysis

LRA Quantifier Elimination

$$(\exists x.(x \geq 2y + z) \wedge (x \leq 3z + 5)) \xrightarrow{QE} (2y - 2z - 5 \leq 0)$$

- For every formula in LRA, there exists an equivalent quantifier-free formula, also in LRA.
- Algorithms to compute quantifier elimination are very costly (doubly exponential in LRA)

Parameter Decoupling

Idea

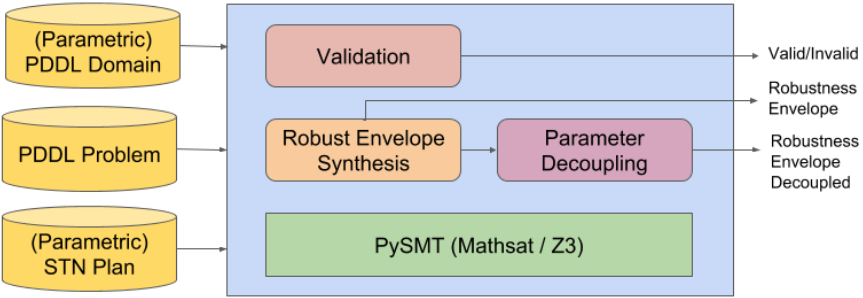
Extract an axis-parallel hyper-rectangle from the robustness envelope to:

1. compactly represent an under-approximation of the parameter space
2. obtain parameter independence from one another

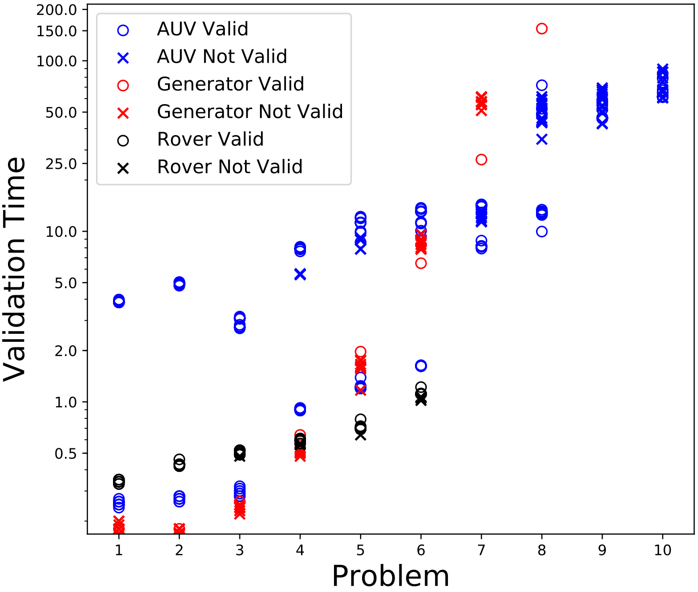
$$
\begin{aligned}
&\mathit{maximize} \quad \sum_{\gamma_i \in \Gamma} (ub_i - lb_i) \quad s.t. \\
&\Big(\bigwedge_{\gamma_i \in \Gamma} lb_i \leq ub_i\Big) \wedge \\
&\forall \bar{\Gamma}.\Big(\Big(\bigwedge_{\gamma_i \in \Gamma} lb_i \leq par_i \leq ub_i\Big) \rightarrow \rho(\bar{\Gamma})\Big)
\end{aligned}
$$

Outline

Implementation



Validation of STN Plans



Synthesis of Envelopes: Impact of Problem Size

Problem	1	2	3	4	5	6
AUV	9.8	16.4	25.6	21.7	33.9	60
Generator	0.31	0.28	0.46	1.12	23.1	Time Out
Solar Rover	0.75	1.03	1.39	1.64	2.25	3.45

Synthesis of Envelopes: Impact of Number of Parameters

Problem	1	2	3	4	5	6
AUV_#1	1.7	0.78	0.97	3.14	51.15	TO
AUV_#2	2.92	1.05	1.32	7.41	94.84	TO
AUV_#3	5.1	1.2	1.82	9.87	107.17	TO
AUV_#4	7.06	1.2	2.04	16.36	89.1	TO
Gen_#1	11.14	59.91	542.3	6350.3	TO	TO
Gen_#2	14.13	72.76	615.22	TO	TO	TO
Gen_#3	375.4	422.55	1130.43	TO	TO	TO
Gen_#4	TO	TO	TO	TO	TO	TO
Rover_#1	1.59	2.32	3.83	5.55	5.28	8.47
Rover_#2	2.69	4.52	5.14	5.62	8.32	13.02
Rover_#3	6.49	6.67	9.07	7.98	11.55	19.7
Rover_#4	8.0	32.72	22.16	12.52	67.6	29.55

Outline

Conclusions

Summary

1. Validate STN plans in action-based setting (full PDDL 2.1)
2. Definition and formalization of robustness envelopes synthesis
3. Parameter decoupling
4. Initial implementation and experiments

Future Directions

1. Scalability! Maybe use approximated quantifier elimination
2. Theoretical and practical comparison with Strong Temporal Planning with Uncontrollable Durations
3. Exploit robustness enveloped in execution (beyond simple STN)

Thanks for your attention!

Backup Slides

Backup 1

TODO